#include<iostream>

#include<string>

#include<algorithm>

#include<queue>

#include<vector>

#include<sstream>

#define MAX 35

using namespace std;

int inorder[MAX], postorder[MAX];

struct Bitree {

int key;

Bitree\*LeftC;

Bitree\*RightC;

};

Bitree \*T = NULL;

int findindex(int\*array,int key)

{

for (int i = 0; i < MAX; i++)

{

if (array[i] == key)

return i;

}

return -1;

}

void cons\_the\_tree(Bitree \*&t, int postindex, int inorderindex, int len)

{

if (len <= 0)

{

t = NULL;

return;

}

else

{

Bitree \*temp = (Bitree\*)malloc(sizeof(Bitree));

t = temp;

t->key = postorder[postindex];//generate the root node;

int root\_in\_inord = findindex(inorder, t->key);

int leftsublen = root\_in\_inord-inorderindex;

cons\_the\_tree(t->LeftC, postindex-(len - leftsublen - 1)-1, inorderindex, leftsublen);

int rightsublen = len - leftsublen - 1;

cons\_the\_tree(t->RightC, postindex - 1, root\_in\_inord + 1, rightsublen);

}

}

//PostInCreateTree(T->lchild,PostIndex - (subTreeLen - 1 - LenF) - 1,InIndex,LenF);

void printinlevelorder()

{

queue<Bitree\*>pushqueue;

pushqueue.push(T);

int flag = 1;

while (!pushqueue.empty())

{

Bitree \*temp = pushqueue.front();

pushqueue.pop();

if (temp == NULL)continue;

if (flag)

{

cout << temp->key;

flag = 0;

}

else

{

cout << " " << temp->key;

}

pushqueue.push(temp->LeftC);

pushqueue.push(temp->RightC);

}

}

int main()

{

int num;

cin >> num;

for (int i = 0; i < num; i++)

{

cin >> postorder[i];

}

for (int i = 0; i < num; i++)

cin >> inorder[i];

cons\_the\_tree(T, num - 1, 0, num);

//output

printinlevelorder();

return 0;

}